International Journal of Neural Systems, Vol. 20, No. 4 (Aug, 2010) 267–278

LEARNING EYE VERGENCE CONTROL FROM A DISTRIBUTED DISPARITY REPRESENTATION

NIKOLAY CHUMERIN[1], AGOSTINO GIBALDI[2], SILVIO SABATINI[2], MARC M. VAN HULLE[1]
[1]*Laboratorium voor Neuro- en Psychfysiologie, Medical School, Katholieke Universiteit Leuven*
Campus Gasthuisberg O&N2, bus 1021, Herestraat 49, 3000 Leuven, BELGIUM
[2]*Department of Biophysical and Electronic Engineering, University of Genoa*
Via all'Opera Pia 11/A, 16145 Genova, ITALY
{*Nikolay.Chumerin, Marc.VanHulle*}*@med.kuleuven.be,* {*Agostino.Gibaldi, Silvio.Sabatini*}*@unige.it*

We present two neural models for vergence angle control of a robotic head, a simplified and a more complex one. Both models work in a closed-loop manner and do not rely on explicitly computed disparity, but extract the desired vergence angle from the post-processed response of a population of disparity tuned complex cells, the actual gaze direction and the actual vergence angle. The first model assumes that the gaze direction of the robotic head is orthogonal to its baseline and the stimulus is a frontoparallel plane orthogonal to the gaze direction. The second model goes beyond these assumptions, and operates reliably in the general case where all restrictions on the orientation of the gaze, as well as the stimulus position, type and orientation, are dropped.

Keywords: vergence control, distributed disparity, convolutional network.

1. Introduction

Vergence eye movements have the task to align both the left and the right eyes on the same object, in order to allow the fusion of the binocular image, and thus to produce singleness of vision. Since the eyes are located in slightly different viewpoints, one object in the word projects on the retinas in different positions, and this difference is defined retinal disparity, that is the cue used for vergence. In fact, both eyes rotate in opposite directions accordingly to the retinal disparity: crossed disparities elicit convergence and uncrossed disparities elicit divergence, so as to achieve and/or maintain the singleness of vision. In this study, we consider a stereo setup consisting of a fixed robotic head with a pair of eyes (see Fig. 1). The task is to estimate, and then to maintain the vergence angle that brings the fixation point along the gaze direction onto the surface of the observed object.

Most of the classic vergence control models [1–6], use as input the *target disparity*, which is defined as the difference between the desired and the actual vergence angles. In this work, similarly to [7–10], we do not use the target disparity, but the foveal images of the eyes as input to the vergence control model.

Theimer and Mallot [7] use a multiscale phase-based approach to compute dense disparity maps. The vergence is adapted in order to minimize the global disparity, albeit that the system fixates at the "average" depth of the scene. Hansen and Sommer in [8] follow the same multiscale approach to estimate the horizontal disparity map. The median disparity of the central area of the disparity map is then used for an asymmetrical vergence control. Stürzl and colleagues in [9] also compute the full disparity map, using responses of complex (position-shift type) horizontal disparity-tuned neurons, for a symmetrical vergence control. In [10], a hierarchical segmentation of the stereo image is computed prior to the estimation of the disparity map, which is then used for a combined vergence/version control. The object nearest to the head is selected as an object of interest (disparity of which is to be nullified).

In this study, we focus on the angular vergence control (in terms of desired vergence angle) rather than kinematic vergence control (in terms of rotational speed of the eyes). As a reason for choosing this type of vergence control is rather practical: in real robotic setups, the precision of the camera rotation speed sometimes is limited (*e.g.*, on a temporal scale), and the angular control offers a better solution.

The above mentioned differences (images instead of disparities as input and angular instead of kinematic vergence control) do not allow for a direct comparison of the models proposed in this article with the standard kinematic vergence control models. Nevertheless, it is worth to mention that the proposed models can be easily adapted to kinematic vergence control models, which we consider as a future step in our research.

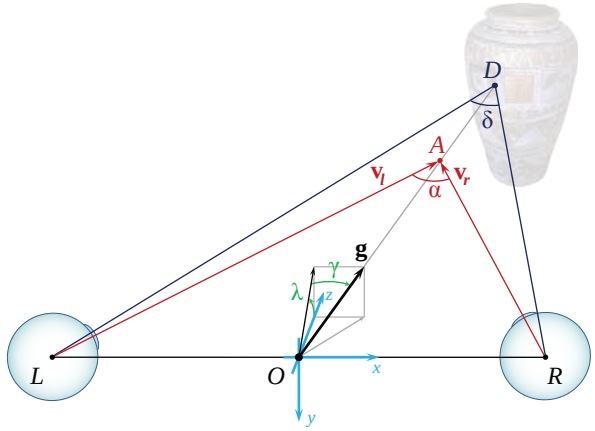


Figure 1: The geometry of the robotic head model. L and R are the nodal centers of the eyes, O is the middle-point of the baseline LR; A is the *actual fixation point*, $|OA|$ is the *actual distance* and α is *actual vergence angle*; D is the *desired fixation point*, $|OD|$ is the *desired distance* and δ *desired vergence angle*. The *gaze direction* is defined as the direction from point O to the fixation point A (and/or D), and depicted by a unit vector $\mathbf{g}$, which, in a headcentric coordinate system $Oxyz$, is specified by a pair of angles γ (pan/yaw) and λ (tilt/elevation). The orientation of the left and right eye visual axes is specified by the vectors $\mathbf{v}_l = \overrightarrow{LA}$ and $\mathbf{v}_r = \overrightarrow{RA}$ respectively.

Concerning the *population coding hypothesis* [11], it consists of the assumption that a distributed disparity representation can be used to remove the ambiguities in the stimulus, and thus to produce the perception of depth by pooling excitations of different disparity detectors. The perception of depth, as well as vergence eye movements, are based on the same visual information extracted at an early stage of cortical processing, the primary visual area (V1), but combined through different specialized mechanisms. In fact, a stimulus consisting of a Random Dot Stereogram (RDS) elicits a response of the disparity-selective neurons in V1. From a behavioural point of view this stimulus is equally effective to provide depth perception and to trigger the correct vergence eye movements. On the contrary, an anti-correlated RDS is ineffective to produce a sensation of depth because the matching process does not find a consistent solution, but triggers a vergence movement in the opposite direction. This is consistent with the experimental evidences reported in [12] of the responses of V1 neurons that, under anti-correlated RDS stimulation, exhibit inverted tuning curves, as predicted by the binocular energy model (see Section 2.4). This suggests that disparity-vergence responses might follow a fast reactive stream that directly involves V1 cells without resorting to a high level interpretation of depth. Further experimental evidences supporting the population coding hypothesis, are presented by [13], where the vergence control is computed by weighting the units' excitations with their preferred disparity. In [14] the global disparity is not computed, but a vergence-related population decoding strategy is used. In particular, the units' excitations are weighted no more with their preferred disparity, but with a set of weights especially designed to derive proper disparity vergence responses.

In this paper, we investigate two vergence control models, based on the population coding hypothesis, that obtain the desired vergence angle from the population response, without explicit calculation of the disparity map. Specifically, here the weights are learnt directly from the desired vergence behaviour, using linear- and non-linear network paradigms. Our experiments show that a linear model can produce an accurate vergence for a *simplified* experiment (fronto-parallel planar stimuli, allowed to move only in the Z-axis direction). Unfortunately, in the *general case*, where restrictions on the stimuli are dropped, the linear model often produces biased results. This fact motivated us to investigate a second model, which relies on a convolutional neural network for the mapping of the disparity population response to the desired vergence angle.

2. Methods

2.1. *Vergence control framework*

For the vergence control paradigm modeling, we have used the framework shown in Fig. 2. This setup consists of the *vergence simulator* module, the *disparity detector population* module, the *population response post-processing* module and the *vergence control network* (VC-net) module.

The main goal of the vergence simulator is to generate stereo image (left and right eye views) based on the actual state of the robotic head: the *vergence*

angle and the *gaze direction* (see Fig. 1), and information about the 3D environment.

The stereo image generated by the simulator is processed by the *disparity detector population*, to produce the population response. Depending on which vergence control network is used, the population response is then directed to either the *population response post-processing* block, which is producing the *post-processed population response* (the linear VC-net case), or directly to the *vergence control network* module (the convolutional VC-net case). The (raw/post-processed) *population response*, together with the actual values of the *gaze direction* and the *vergence angle*, are fed into the *vergence control network* module, the main module of the model. The goal of the VC-net is to produce a new vergence angle, to get the fixation point onto the surface of the object of interest, without changing the gaze direction.

2.2. *Vergence database and training*

For training of the VC-net, we have prepared a *vergence database*. The database consists of two tables: a table of synthetic scenes and a table of vergence samples (see Fig. 3). For efficient memory usage, the scenes were allowed to be reused in several vergence samples. There are two types of synthetic scenes in the vergence database, which correspond to the simplified- and general case scenarios. The simplified case scenes contain only one type of object-stimulus, a fronto-parallel rectangular patch perpendicular to the gaze direction in the primary position (see Fig. 5a). The stimulus in this case is large enough to completely cover the field of view of both cameras.

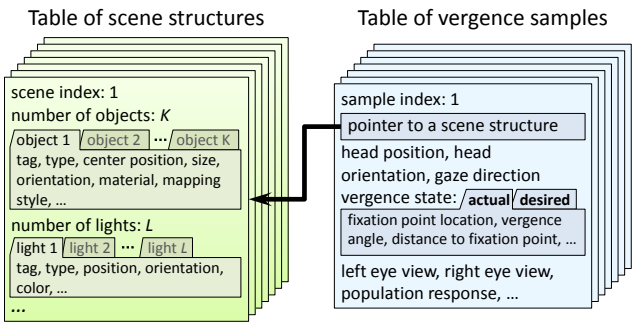


Figure 3: Schematic structure of the vergence database.

The general case scenes consist of several simple (plane rectangular patch, cube, pyramid, tetris-like etc.) textured objects, randomly placed into a room-like virtual environment with several light sources (see Fig. 5b). The object sizes are chosen randomly allowing for depth discontinuities.

Vergence samples consist of the *gaze direction*, the *actual vergence angle*, the *stereo pair* (left and right eyes images), the *population response* for the stereo pair and the *desired vergence angle*. The actual vergence angle is a distorted (with Gaussian noise) version of the desired one. The actual vergence angle is expected to become as close to the desired vergence angle as possible, when running the control model.

Each vergence sample in the database can be considered as a training pair. The input part is constructed from the post-processed (or raw) population response, the gaze direction and the actual vergence angle; the output consists of only one scalar parameter – the desired vergence angle. The vergence database used for the VC-net training consists of 1000 synthetic scenes and 5000 samples. The balance between general and simplified scenes (as well as for the samples) has been set to 50%/50%. The real-world images were used as textures for the objects. To reduce the influence from the possible overfitting to particular textures on the results of the evaluation, we have used non-overlapping sets of textures for the training- and test experiments. An early stopping technique (with 10% of the training data for validation) was used to prevent overfitting during training. To achieve a fair comparison, both VC-nets were trained using the same training data.

2.3. *Vergence simulator*

The vergence *simulator* module consists of the *renderer* and the ideal *robotic head model* (RHM) with fixed neck. In this model, the robotic head is assumed to be fixed and the eyes to rotate around their nodal points. We selected this model because it is easy to implement, and eventually to replace by a real tilt-pan stereo setup.

Given a RHM baseline b, *i.e.*, the distance between the nodal points, the gaze direction is defined by γ and λ, the pan/yaw and tilt/elvation angles, considering the coordinates centered in the cyclopean point O in the middle of the baseline b, as in [15].

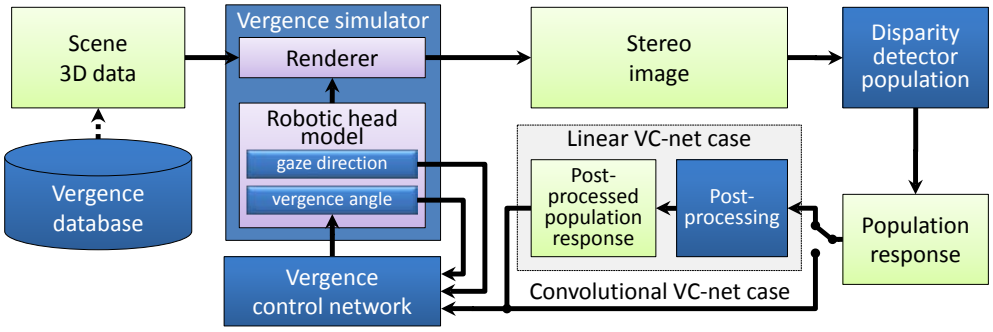


Figure 2: The block diagram of the framework used in vergence control model training and testing (see text).

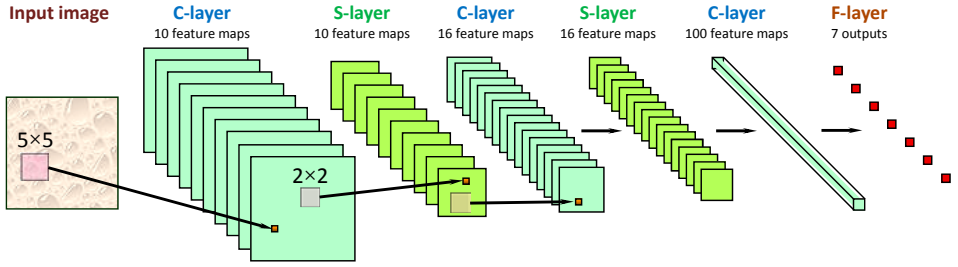


Figure 4: Typical architecture of a convolutional neural network.

The actual vergence angle α is defined as the angle between the left and right visual axes $\mathbf{v}_l$ and $\mathbf{v}_r$ (see Fig. 1). In this study we used the same parameters of the RHM as in [14] (the baseline $b = 70$ mm, the focal length $f_0 = 17$ mm and field of view $\approx 20°$). The RHM module takes as input the vergence angle and the gaze direction to produce the exact position and orientation of both eyes/cameras, needed for the renderer.

The renderer, in turn, produces the stereo image, observed by the left and right eyes (see Fig. 5), using the position/orientation of the eyes, and the geometric description of the scene, provided by the *scene 3D data* block.

To make sure that the disparities are not too large and can be properly handled by the disparity detector population, we decided to render the retinal projections with low resolution *i.e.*, we obtain images of 41×41 pixels for a field of view of $\approx 20°$. Note that the resolution could be higher, but consequently to allow the population to cope with the same range of disparities, the receptive fields of the disparity detectors should be larger, which would significantly increase the computational cost sand thus slow down the simulations.

2.4. *Disparity detectors population module*

Disparity information can be extracted from a stereo image pair by using a distributed cortical architecture [16] that resorts to a population of simple and complex cells. The population is composed of cells sensitive to $N_o \times N_p$ vector disparities $\boldsymbol{\delta} = (\delta_H, \delta_V)$ with N_p magnitude values distributed in the range $[-\Delta, \Delta]$ pixels and along N_o orientations, uniformly distributed between 0 and π (see Fig. 6b). Each simple cell has a binocular receptive field $g_L(x, y) + g_R(x, y)$ defined by a pair of Gabor functions:

$$g(x, y; \psi, \theta) = e^{-(x_\theta^2 + y_\theta^2)/2\sigma^2} \cos(2\pi k_0 x_\theta + \psi) \quad (1)$$

positioned in the corresponding points $\mathbf{x} = (x, y)$ of the left and the right images, rotated by the same angle θ with respect to the horizontal axis, and characterized by the same peak frequency k_0 and spatial envelope σ, and by a proper binocular phase shift $(\Delta\psi = \psi_L - \psi_R)$, along the rotated axis x_θ, from

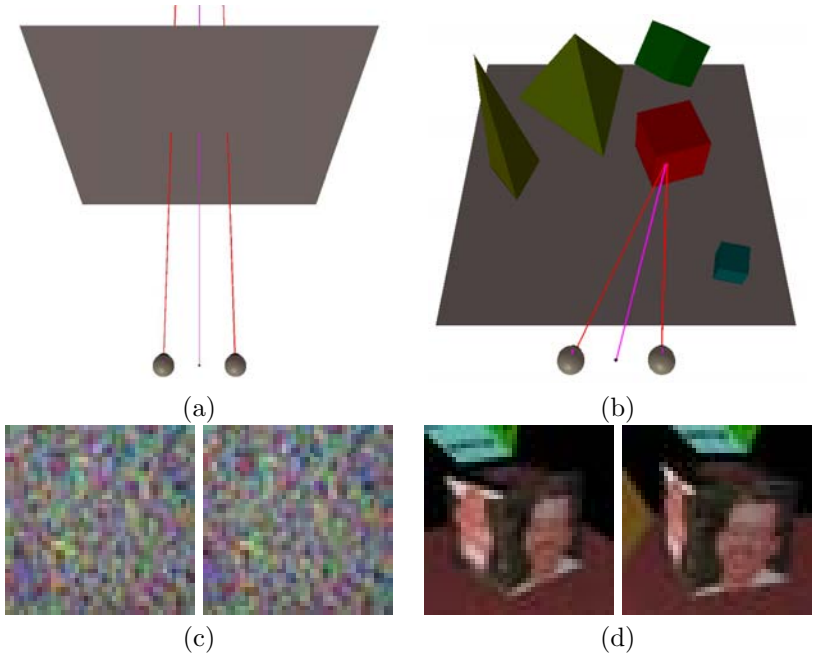


Figure 5: Examples of simplified (a) and general case (b) synthetic scenes used by the simulator to render the corresponding stereo images (c,d).

which the cell obtains its specific tuning to a disparity $\delta_{\text{pref}} = \Delta\psi/2\pi k_0$, along the direction orthogonal to θ. Formally, given $I_L(\mathbf{x})$ and $I_R(\mathbf{x})$, the left and the right images, and $\boldsymbol{\delta}(\mathbf{x})$, the image disparities, so that $I_L(\mathbf{x}) = I_R(\mathbf{x} + \boldsymbol{\delta}(\mathbf{x}))$, for every image position $\mathbf{x}$, the response of a simple cell r_s is given by:

$$r_s(\boldsymbol{\delta}(\mathbf{x}); \theta, \Delta\psi) = \iint \big(g_L(\mathbf{x}' - \mathbf{x}) I_R(\mathbf{x}' + \boldsymbol{\delta}(\mathbf{x}')) + \\ + g_R(\mathbf{x}' - \mathbf{x}) I_R(\mathbf{x}')\big) d\mathbf{x}'. \quad (2)$$

The response of a complex cell r_c is modeled by a sum of the squared responses of a quadrature pair of simple cells (see Fig. 6a), and is given [17] by:

$$r_c(\boldsymbol{\delta}(\mathbf{x}); \theta, \Delta\psi) = r_s^2(\boldsymbol{\delta}(\mathbf{x}); \theta, \Delta\psi) + \\ + r_s^2(\boldsymbol{\delta}(\mathbf{x}); \theta, \Delta\psi + \pi/2). \quad (3)$$

For each orientation, we consider N_p phase-shifts $\Delta\psi$ uniformly distributed between $-\pi$ and π, so that each cell produces a response map r_c^{ij} of the same size $(n_r \times r_c)$ as the binocular image. The population is, in this way, fig:Gabor-filter-bankcapable of providing reliable disparity estimates for each orientation, in the range between $-\Delta$ and Δ, where $\Delta = \Delta\psi_{\max}/k_0$ can be defined as the maximum detectable disparity of the population.

In this work, we consider only a single-scale architecture of the disparity detector population, but the population can be readily extended to the multiscale mode, without conceptually changing our framework, but which will be computationally much more expensive.

2.5. *Post-processing module*

The post-processing of the population response is used only for the linear VC-net, and comprises a two-dimensional convolution over the first two (spatial) dimensions of the population response, using a two-dimensional Gaussian kernel G_σ:

$$P_{ij} = G_\sigma * r_c^{ij}, \quad (4)$$

where r_c^{ij} is the population response map for the i-th orientation and the j-th phase shift. The kernel G_σ has the same size $n_r \times n_c$ as the size of a population response map r_c^{ij}, so the result of the convolution is a scalar value P_{ij}.

On the one hand, this step drastically reduces the amount of data to further process. Indeed, after pooling, the network has to process only a two-dimensional $(N_o \times N_p)$ pooled population response instead of a four-dimensional $(n_r \times n_c \times N_o \times N_p)$

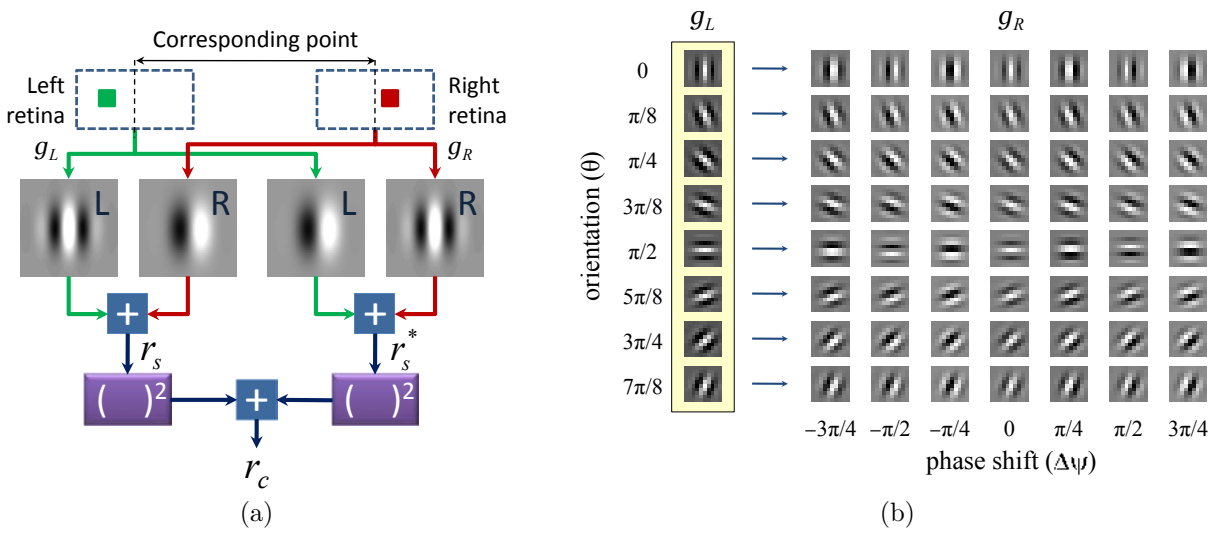


Figure 6: (a) The construction of the complex cell response r_c as a sum of squared responses of a quadrature pair of simple cells, where $r_s^* = r_s(\boldsymbol{\delta}(\mathbf{x}); \theta, \Delta\psi + \pi/2)$. The green and red pathways relate to the monocular "quadrature pair" of simple cell receptive fields, g_L and g_R, respectively. (b) The Gabor filter bank used in the population with binocular receptive fields for each retinal location.

array, where N_p is the number of phase shifts, and N_o the number of orientations. But, on the other hand, the pooling has a major drawback as it discards the spatial information about the disparity encoded in the population response. The results of simulations (see Section) revealed that, in the general case scenario, this discarding could lead to degraded vergence accuracy.

The convolutional network works directly on the population response, and the post-processing is done in the first two layers of the convolutional network.

2.6. *Vergence control module*

This module is the main module of the model. The purpose of it is to convert the post-processed population response together with the actual vergence, and the gaze direction, into a new vergence angle. Virtually, this module can be represented by any kind of paradigm, but in this work we discuss only a linear network and a convolutional network.

2.7. *Linear network*

The first attempt in developing a network model for vergence control was with the simplest possible solution consisting of only a single linear unit (see Fig. 7).

The simulations revealed (see Section 3) that even this simple network is able to produce accurate angular vergence control in some restricted situations (*e.g.*, in the simplified case). The input vector for the linear VC-net was constructed as a concatenation of the pooled population response (56 values), the gaze direction (2 values) and the actual vergence (1 value), so its dimensionality is 59. The output is a prediction of the vergence angle, which is a scalar value. Due to the linearity of the network, there was no reason to introduce any hidden layers, so the linear VC-net consisted of only one linear unit. This simplest possible vergence control network has only 60 parameters (including bias), which can be learned either directly (using linear regression or its robust modification), or iteratively (using gradient descent), from the training database.

2.8. *Convolutional network*

Convolutional networks (CNs) appeared in the 80s and became popular in Computer Vision [18–20] mainly due to efforts of Yann LeCun and coworkers [21]. All CNs have common architectural features: *local receptive fields*, *shared weights*, and spatial or temporal *subsampling*, which allow them to achieve some degree of shift and deformation in-

variance and, at the same time, reduce the number of training parameters.

A typical convolutional network is a feed-forward network of layers of three types: *convolutional* (C-layer), *subsampling* (S-layer) and *fully-connected* (F-layer). The C-layers and S-layers usually come in pairs and are interleaved, and F-layers come at the end (see Fig. 4). The output of a C-layer is organized as a set of *feature maps*. Each feature map contains the output of a set of neurons with local receptive fields. All neurons in the feature map share the same weights, so the feature map is responsible for a particular local visual feature, encoded in the weights of these neurons. The computation of a feature map starts with a 2D convolution of the input with a fixed kernel defined by the neuron's weights. A feature map can have inputs from several feature maps of the previous layer. In order to condense the extracted features, and to make them more invariant with respect to spatial deformations, the C-layer is typically followed by an S-layer which performs a local averaging and subsampling. Each neuron in a F-layer just adds a bias to the weighted sum of all inputs and then propagates the result through a nonlinear transfer function (RBF or sigmoid).

The network is trained in a supervised manner using backpropagation. For the efficient training of large CNs, LeCun and colleagues proposed a modification of the Levenberg-Marquardt algorithm [22].

The architecture of the convolutional network, used for our experiments is shown in Fig. 8. The main challenge in this approach was the amount of data: the population response consists of 56 (8×7) maps of resolution 41×41 (rendered image resolution), so the input of the network has 94136 ($41 \times 41 \times 8 \times 7$) components. In order to be able to train the network with such high dimensional input data, we had to reduce the number of training parameters. The first (convolutional) layer is a fixed set of (nontrainable) Gaussian kernels of size 19×19 with standard deviation 6. The second (subsampling) layer has also 56 feature maps size of which was set to 3×3.

2.9. *Vergence performance measures*

Given the RHM, from the gaze direction vector $\mathbf{g}$, ($\|\mathbf{g}\| = 1$), it is possible to infer the actual distance $d = |OA|$ to the fixation point A from the middle of the head's baseline O using the actual vergence angle α (see Fig. 1):

$$\begin{aligned} d &= \frac{b}{2}\left(s + \sqrt{s^2+1}\right), \text{ where} \\ s &= \cot\alpha\cos\gamma \end{aligned} \tag{5}$$

and *vice versa*:

$$\begin{aligned} \alpha &= \arccos\left(\frac{\mathbf{v}_l^T\mathbf{v}_r}{\|\mathbf{v}_l\|\cdot\|\mathbf{v}_r\|}\right), \text{ where} \\ \mathbf{v}_l &= d\cdot\mathbf{g} + (b/2,0,0)^T, \text{ and} \\ \mathbf{v}_r &= d\cdot\mathbf{g} - (b/2,0,0)^T. \end{aligned} \tag{6}$$

where $\mathbf{v}_l$ and $\mathbf{v}_r$ are the visual axes of respectively the left and right eye. From the equations (5) and (6), one can see that, by considering a fixed gaze direction $\mathbf{g}$ and a fixed baseline b, the vergence angle $\alpha \in (0;\pi)$ can be diffeomorphically mapped into the distance to the fixation point d (nevertheless the mapping is nonlinear).

In our experiments $\alpha \in (4°, 10°)$ and, as it follows from (5), even for a small vergence angle α (more distant stimulus), the deviation leads to a significant change of d. In this case, the deviation of the actual distance to the fixation point d from the desired one, more adequately reflects the accuracy of the vergence model, than the deviation of the corresponding vergence angles. Due to this anisotropy of the distance uncertainty, we prefer the distance-based measure for the assessment of the model performance over the angular-based.

2.10. *Experiments*

To evaluate both VC-nets, as already mentioned, we consider two cases for the experiment: a *simplified* and a *general* case. An example of the simplified case is shown in Fig. 5(a,c): the gaze direction of the robotic head is orthogonal to its baseline, and the stimulus is in the frontoparallel plane which is also orthogonal to the gaze direction. In the general case, all restrictions on the orientation of the gaze, as well as the stimulus position, type and orientation, are dropped. One of the examples is shown in Fig. 5(b,d).

A series of 100 vergence maintenance experiments have been carried out for both VC-nets, for both scenarios. Each experiment consisted of 100 steps

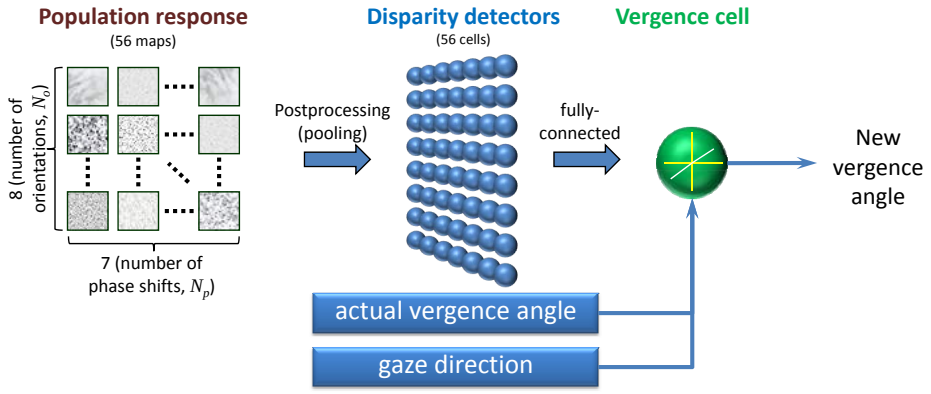


Figure 7: Linear VC-net and its inputs.

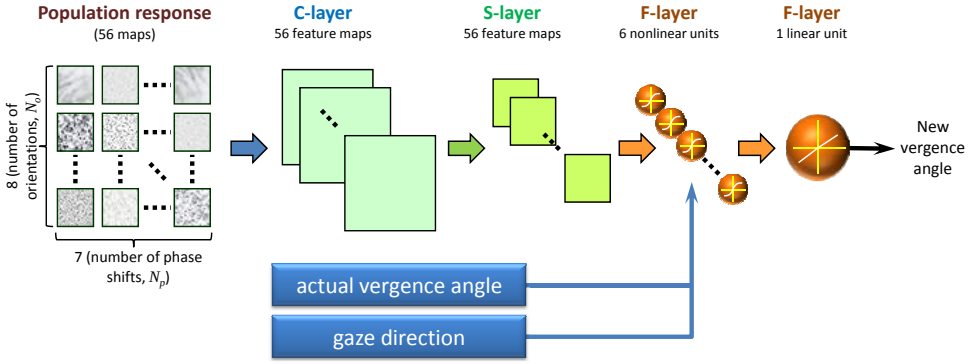
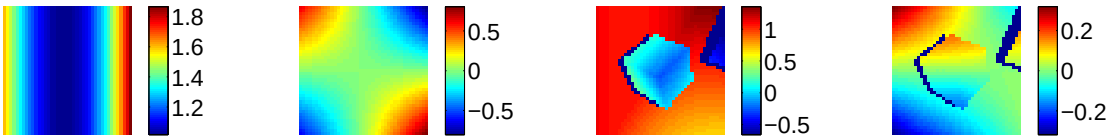


Figure 8: Convolutional VC-net and its inputs.



(a) (b) (c) (d)

Figure 9: Typical examples of horizontal (a,c) and vertical (b,d) disparity maps for the simplified (a,b) and general case (c,d) synthetic scenes. In the simplified case (a,c), the disparity maps have the same symmetrical patterns, and differ only by the magnitude of the disparity. In the general case (b,d), the disparity maps usually have discontinuities, and are not symmetrical.

during which the randomly generated stimulus was moving along the gaze direction, changing its distance (from 400 mm to 900 mm) to the head in a particular manner. We have considered three patterns of the stimulus motion-in-depth: ramp, sinusoid and staircase. Pretrained VC-nets were allowed to control the actual vergence angle to keep the fixation point as best as possible on the surface of the stimulus. During each experiment, the actual and the desired values of the vergence angle, and the distance to the stimulus were stored for each time step, for further analysis.

3. Results

The results of the evaluation experiments described in Section 3 of both VC-networks in both considered scenarios are presented in Fig. 10 and Table 1. Each panel of Fig. 10 contains 1) the desired (ground truth) distance to the stimulus curve depicted by the solid green curve, 2) the mean (averaged across all experiments) actual distance to the stimulus curve depicted by the dashed red curve, and 3) the variance (standard deviation across all experiments) of the actual distance margins depicted by the dotted black curve.

The performance of the VC-net can also be assessed using the ratio of the distance-based error variance to the corresponding desired distance. The less this ratio is, the lower relative (distance) error is produced by the network. Table 1 contains the minimal, mean, median and maximal values of this ratio (in percent) for each experiment type and each stimulus.

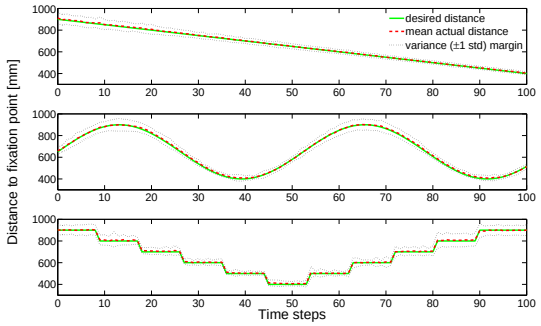
4. Discussion

From Fig. 10(a,c) and Table 1, it is clear seen that both networks performs relatively well in the simplified scenario: the mean actual distance curve almost coincides with the desired one, and the variance in both cases is relatively small. For the general case scenario, the situation is different. The linear VC-net (Fig. 10b) shows a much larger variance and a general tendency to over(under)shoot towards the "average" depth of the scene (at approximately 600 mm). The convolutional VC-net (Fig. 10d) also shows a relatively larger variance, but the mean actual distance is closer to the ground truth than in the linear VC-net case. The effect of the anisotropy of the distance uncertainty, mentioned in Section 2.9, is noticeable in Fig. 10: the further the stimulus is, the larger mistakes made by the VC-net.

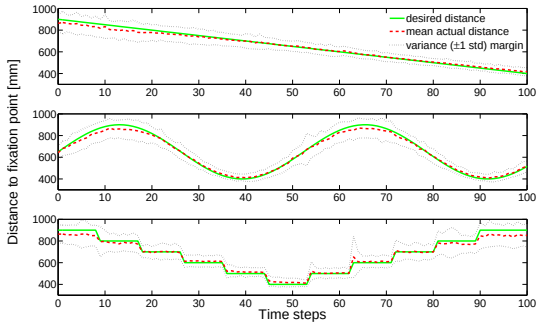
The larger magnitude of the vergence error of the linear network in the general case, with respect to simplified case, can be explained, from our point of view, mainly by the disparity discontinuities, and possibly by the presence of the vertical disparity asymmetric patterns. The disparity discontinuities are usually caused by the limited size of the stimuli, which do not entirely cover the field of view in both eyes, or by the non-convex shape of the stimuli (*e.g.*, tetris-like objects). The horizontal and vertical disparities in the simplified case (see Fig. 9(a,b)) have very simple symmetrical patterns. While in the general case, these patterns are not so simple and usually not symmetrical (see Fig. 9(c,d)). This irregularity of the disparity is caused by the arbitrary orientation and location of the object surface, as well as by the depth discontinuities, and the not always convex shape of the stimulus-object.

For the linear VC-net, in the simplified scenario, the vertical disparity is symmetrically spread over the spatial dimensions of the population response, and is discarded in the preprocessing stage, due to the spatial pooling. This does not always happen in the general case, so the pooled population response is biased by the residual vertical disparity, which in turn leads to a bias in the vergence angle, at convergence. This situation motivated us to investigate a more complex paradigm for vergence control, one which should be able to recognize particular patterns in the population responses in the general case, and produce a proper vergence control signal. The idea behind the use of the convolutional network, as a vergence controller, relies on the assumption that this powerful network, after proper training, will be able to recognize disparity patterns directly from the population responses, and convert them into the desired vergence angle.

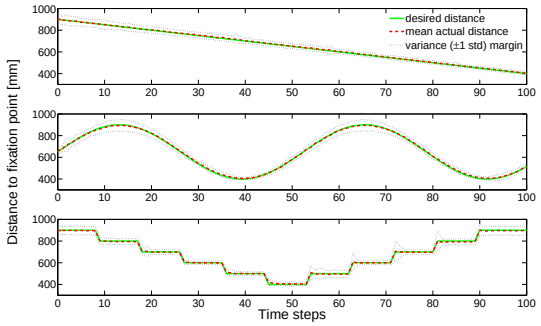
There is also an interesting effect which we discovered during testing: when the stimulus is too far from the actual fixation point, leading to too large disparities for the population to handle, both networks, in the majority of the cases, choose the *proper direction* of the vergence. In this case, the fixation point will not land on the surface of the stimulus-object after the first iteration, but after a few more



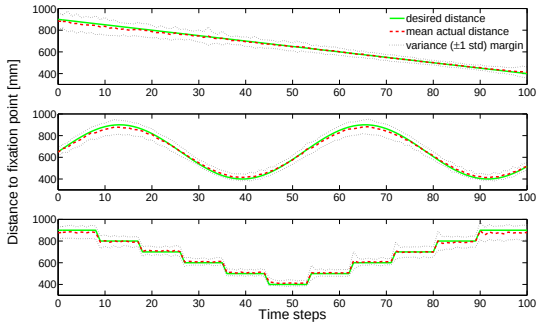
(a) Linear VC-net, simplified scenario.



(b) Linear VC-net, general case scenario.



(c) Convolutional VC-net, simplified scenario.



(d) Convolutional VC-net, general case scenario.

Figure 10: Results of the depth-based performance plots for both VC-nets in both scenarios.

iterations. This effect can explain the larger error variation for the staircase stimulus (see Fig. 10, third plot in each panel) comparing to the remaining stimuli: the VC-net is not able to handle large distance steps in one-shot manner and extra iterations are needed. The first iteration in this case systematically undershoots causing larger distance error, which explains the "spikes" in the error variance curves around the depth steps (in staircase experiments).

We also should discuss some limitations of the proposed models. As our models heavily depend on the quality of the population response, all the limitations of the local distributed disparity methods (poor performance on homogenous textures, short range of the disparity) apply to the proposed models as well. To reduce the effect resorted by these limitations, we suggest to avoid large objects with homogenous textures, and to replace the filters in the population by ones with a larger support, in order to tackle large disparities.

Both proposed models use iterative training based on input images and, therefore, there is a possibility that training will overfit the models on representing the textures used during training. Unfortunately, completely eliminating this possibility, as well as to prove the opposite statement (about the independence from textures), is not possible. One way to avoid this problem, is to consider a large variety in textures in the training set.

5. Conclusion

Most of the conventional vergence control models [1–7, 23, 24], are based on the minimization of the horizontal disparity. Following an approach similar to [25], we propose to avoid the explicit computation of the disparity map, and to extract the desired vergence angle directly from the population response, over the "foveal" region, of a cortical-like network organized as hierarchy of arrays of binocular complex cells [16]. A neural network paradigm has been chosen for this type of conversion/extraction procedure. Although the paradigm only resorts to a population of neurons in a single scale, we demonstrate that, using a neural network paradigm, accurate and fast

Table 1: Variance of distance-based error relatively to desired distance.

VC-net	Experiment scenario	Stimulus type	Error variance ratio (%)			
			min	mean	median	max
Linear	Simplified case	Ramp	2.6828	3.8921	3.6172	6.2715
		Sinusoid	2.8904	5.2275	5.2840	7.5772
		Staircase	2.6566	5.4345	5.2589	10.6765
	General case	Ramp	6.4996	8.4466	8.1057	12.9288
		Sinusoid	6.2622	10.0322	9.9448	22.1558
		Staircase	7.1387	10.6848	9.9420	31.9260
Convolutional	Simplified case	Ramp	2.4841	3.7045	3.6237	5.8980
		Sinusoid	2.2913	4.8870	4.9034	8.6034
		Staircase	2.1722	4.3578	3.6502	13.2121
	General case	Ramp	4.0339	6.4378	5.7930	12.7739
		Sinusoid	4.8622	6.9828	6.8680	10.6242
		Staircase	3.9617	6.5304	6.2880	13.7682

vergence control can be achieved in a closed loop, for different orientations of the gaze. Comparison of the performances of the linear and the convolutional VC networks leads us to a conclusion that, in the simplified case, both networks demonstrate very similar performances. Yet, the convolutional VC-net performs better than the linear one in a more general scenario where any assumption on the scene structure and restrictions on the gaze direction are dropped. The improved performances of the convolutional network comes at the price of a higher number of iterations, which, unfortunately, make convolutional networks much more computationally expensive than the linear-based one.

Acknowledgment

This work has been partially supported by the EU Project FP7-ICT-217077.

References

1. G. Westheimer and A.M. Mitchell. Eye movement responses to convergence stimuli. *Archives of Ophthalmology*, 55(6):848, 1956.
2. C. Rashbass and G. Westheimer. Disjunctive Eye Movements. *Journal of Phisyology*, (159):339–360, 1961.
3. V.V. Krishnan and L.A. Stark. A heuristic model for the human vergence eye movement system. *IEEE Trans. Biomed. Eng.*, 24:44–49, 1977.
4. C.M. Schor. The relationship between fusional vergence eye movements and fixation disparity. *Vision Research*, 19(12):1359–1367, 1979.
5. G.K. Hung, J.L. Semmlow, and K.J. Ciuffreda. A dual-mode dynamic model of the vergence eye movement system. *IEEE Trans. Biomed. Eng.*, 36(11):1021–1028, 1986.
6. M. Pobuda and C.J. Erkelens. The relationship between absolute disparity and ocular vergence. *Biological Cybernetics*, 68(3):221–228, 1993.
7. W.M. Theimer and H.A. Mallot. Phase-based vergence control and depth reconstruction using active vision. *CVGIP, Image understanding*, 60(3):343–358, 1994.
8. M. Hansen and G. Sommer. Active depth estimation with gaze and vergence control using gabor filters. In *Pattern Recognition, Proceedings of the 13th International Conference on*, volume 1, pages 287–291, Aug 1996.
9. W. Stürzl, U. Hoffmann, and H.A. Mallot. Vergence control and disparity estimation with energy neurons: Theory and implementation. *Lecture notes in computer science*, pages 1255–1260, 2002.
10. R. Marfil, C. Urdiales, J.A. Rodriguez, and F. Sandoval. Automatic vergence control based on hierarchical segmentation of stereo pairs. *IJIST*, 13(4):224–233, 2003.
11. W. Richards. Anomalous stereoscopic depth perception. *Journal of the Optical Society of America*, 61(3):410–414, 1971.
12. B.G. Cumming and A.J. Parker. Responses of primary visual cortical neurons to binocular disparity without depth perception. *Nature*, 389:280–283, 1997.
13. H.A. Mallot, A. Roll, and P.A. Arndt. Disparity-evoked vergence is driven by interocular correlation. *Vision Research*, 36(18):2925–2937, 1996.
14. A. Gibaldi, M. Chessa, A. Canessa, S.P. Sabatini, and F. Solari. A cortical model for binocular vergence control without explicit calculation of disparity. *Neurocomp.*, 73:1065–1073, 2010.
15. M. Hansard and R. Horaud. Cyclopean geometry of

binocular vision. *J. Opt. Soc. Am.*, 25:2357–2369, 2008.

16. M. Chessa, S.P. Sabatini, and F. Solari. A fast joint bioinspired algorithm for optic flow and two-dimensional disparity estimation. In *Proc. International Conference on Computer Vision Systems (ICVS'09)*, Liege, Belgium, October 2009.
17. N. Qian. Computing stereo disparity and motion with known binocular cell properties. *Neural Computation*, 6(3):390–404, 1994.
18. P.Y. Simard, D. Steinkraus, and J.C. Platt. Best practices for convolutional neural networks applied to visual document analysis. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition (ICDAR'03)*, volume 2, pages 958–962. IEEE Computer Society, 2003.
19. Y. LeCun, F.J. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of CVPR'04*. IEEE Press, 2004.
20. J. Ruiz-Pinales, R. Jaime-Rivas, E. Lecolinet, and M.J. Castro-Bleda. Cursive word recognition based on interactive activation and early visual processing models. *Int J Neur Syst*, 18(5):419–431, 2008.
21. Y. LeCun, L. Bottou, G. Orr, and K. Muller. Efficient backprop. In G. Orr and Muller K., editors, *Neural Networks: Tricks of the trade*. Springer, 1998.
22. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. volume 86, pages 2278–2324, 1998.
23. S.S. Patel, H. Ogmen, and B.C. Jiang. Neural network model of short-term horizontal disparity vergence dynamics. *Vision Research*, 37(10):1383–1399, 1996.
24. J. Horng, J. Semmlow, G.K. Hung, and K. Ciuffreda. Initial component in disparity vergence: A model-based study. *IEEE Trans. Biomed. Eng.*, 45:249–257, 1998.
25. A. Gibaldi, M. Chessa, A. Canessa, S.P. Sabatini, and F. Solari. A neural model for binocular vergence control without explicit calculation of disparity. In *Proc. European Symposium on Artificial Neural Networks (ESANN'09)*, Bruges, Belgium, April 2009.